**Classwork 3**

**Assignment 1:**

Define a procedure which using OpenCV loads an image and shows it on the screen twice. One instance is the image plotted by OpenCV and another one plotted by Matplotlib.

**Assignment 2:**

Define a function **myBGR2RGB** which receives an image in BGR representation and returns it in the RGB representation (you are not allowed to use built in functions for representation switching). Call example: **myBGR2RGB(myImage)**

**Assignment 3:**

Define a procedure which loads image (img4.jpg) splits it into three matrices representing R, G and B and plots these matrices in the same window side by side in the color they represent. Then plots additional window with the image in which B is transposed and G is upside down.

**Assignment 4:**

Define a function which receives an image and draws a green horizontal line in the middle of the image, then returns the resulted image.

**Assignment 5:**

Define a function which receives an image and draws a red diagonal line through the image, then returns the resulted image.

**Assignment 6:**

Define a function which receives an image and draws a blue square in the middle of that image, then returns the resulted image.

**Assignment 7:**

Define a function which receives an image and draws a purple circle in the middle of that image, then returns the resulted image.

**Assignment 8:**

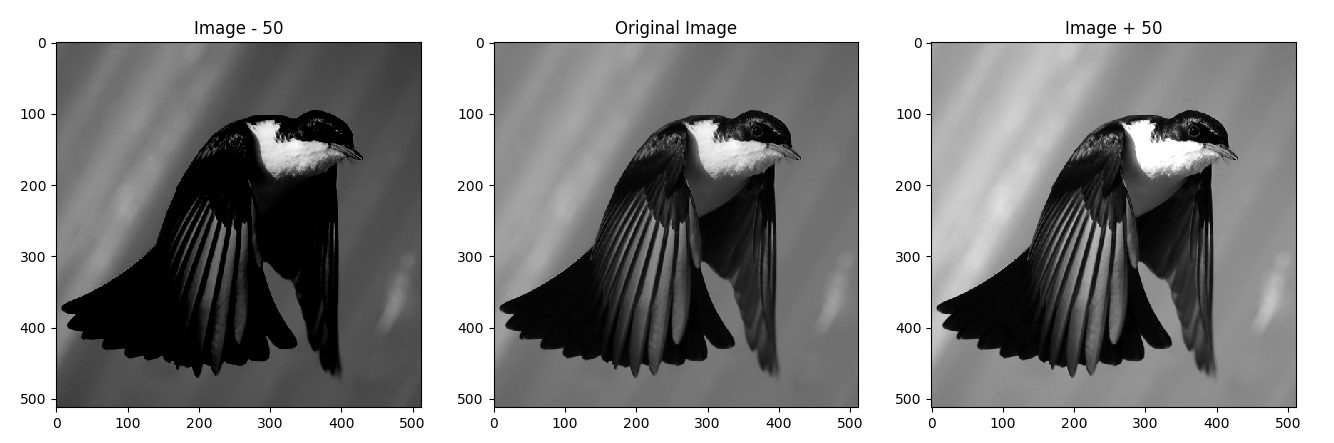
Define a function which receives an image and draws a triangle in the middle. Each side of the triangle should have a different color; red, green or blue.

**Assignment 9:**

Define a procedure which loads an image, turns it into grey scale representation, shows that image and in the same window alongside it shows the same image plus 50 and the same image minus 50 (Image should have 8-bits). Any value above 255 should be replaced with 255 and any value below 0, should be replaced with 0.

Note: Mathematical actions should be done on an appropriate data type.

Result example:



**Assignment 10:**

Define a function **myZeroPadding** which receives an image and a padding size, and returns a zero padded image (padding size represents amount of pixels to be added on each side). Call example: **myZeroPadding(myImage, padSize)**

**Assignment 11:**

Define a function **myExtendedPadding** which receives an image and a padding size, and returns the image with extended borders (border size is the padding size). Call example: **myExtendedPadding(myImage, padSize)**

**Assignment 12:**

Define a function which receives an image, calculates its x-derivative and its y-derivative, calculates a mean matrix of the two derivative results, normalizes it to 8-bit representation, converts it into an 8-bit image with standard representation and returns the resulted image.